The surprizing complexity of generalized reachability games

Nathanaël Fijalkow, Florian Horn

The surprising complexity of generalized reachability games

Nathanaël Fijalkow[1,2] and Florian Horn[1]

[1] LIAFA
CNRS & Université Denis Diderot - Paris 7, France
{nath,florian.horn}@liafa.jussieu.fr
[2] ÉNS Cachan
École Normale Supérieure de Cachan, France

Abstract. Games on graphs provide a natural and powerful model for reactive systems. In this paper, we consider generalized reachability objectives, defined as conjunctions of reachability objectives. We first prove that deciding the winner in such games is PSPACE-complete, although it is fixed-parameter tractable with the number of reachability objectives as parameter. Moreover, we consider the memory requirements for both players and give matching upper and lower bounds on the size of winning strategies. In order to allow more efficient algorithms, we consider subclasses of generalized reachability games. We show that bounding the size of the reachability sets gives two natural subclasses where deciding the winner can be done efficiently.

1 Introduction

Graphs games. Our purpose is to study reactive systems by abstracting them into graphs games: a state of the system is represented by a vertex in a finite directed graph, and a transition corresponds to an edge. If in a given state, the controller can choose the evolution of the system, then the corresponding vertex is controlled by the first player, Eve. Otherwise, the system evolves in an uncertain way: we consider the worst-case scenario where a second player, Adam, controls those states. To a run of the system corresponds a play on the game: we put a pebble in the initial vertex, then Eve and Adam move this pebble along the edges, constructing an infinite sequence. The specification of the system gives an objective Eve tries to ensure on this sequence. In order to synthesize a controller, we are interested in two questions: whether Eve wins in the game, and what resources are needed to construct a winning strategy (see [GTW02] for more details).

System specifications. To specify properties of a system, we construct a set of infinite sequences representing the correct behaviors of the system. From an infinite sequence we extract finite information to decide whether the run

it represents meet the specification. For instance, considering the set of vertices visited infinitely often allows to specify the classical ω-regular properties, *e.g* Büchi, parity, Streett, Rabin and Müller objectives. Other informations can be carried out, as for instance the set of vertices visited with positive frequency [TBG09], or the order in which the vertices are visited for specifying LTL objectives [KPV07,HTW08,Zim11]. In this work, we observe the set of vertices visited at least once, which allows to specify reachability objectives, also called weak objectives [NSW02,SW74,Mos91,KVW00]

Generalized reachability objectives. The (simple) reachability objective requires, given a subset of vertices F, that a vertex from F is reached. Reachability objectives only specifies that one property (represented by F) is satisfied along the run. We allow more properties to be specified by using generalized reachability objectives, defined as conjunctions of k reachability objectives. In this context, a reachability objective is often referred as a color: a generalized reachability objective is then to see each of the k colors at least once.

2 Definitions

The games we consider are played on an *arena* $\mathcal{A} = (V, (V_\circ, V_\Box), E)$, which consists of a finite graph (V, E) and a partition $(V_\circ, V_\Box)$ of the vertex set V: a vertex is controlled by Eve if it belongs to $V_\circ$ and by Adam if it belongs to $V_\Box$. Vertices from $V_\circ$ are depicted by a circle, and vertices from $V_\Box$ by a square. We denote by n the number of vertices and m the number of edges. Playing consists in moving a pebble along the edges: the pebble is placed on the initial vertex v_0, then the player who controls the vertex chooses an edge and sends the pebble along this edge to the next vertex. From this infinite interaction results a *play* π, which is an infinite sequence of vertices $v_0, v_1, \dots$ where for all i, we have $(v_i, v_{i+1}) \in E$, *i.e* π is an infinite path in the graph. We denote by Π the set of all plays, and define *objectives* for a player by giving a set of winning plays $\Phi \subseteq \Pi$. The games are zero-sum, which means that if Eve has the objective Φ, then Adam has the objective $\Pi \setminus \Phi$ (the objectives are opposite). Formally, a *game* is given by a couple $\mathcal{G} = (\mathcal{A}, \Phi)$ where $\mathcal{A}$ is an arena and Φ an objective.

A *strategy* for a player is a function that prescribes, given a finite history of the play, the next move. Formally, a *strategy* for Eve is a function $\sigma : V^* \cdot V_\circ \to V$ such that for a finite history $w \in V^*$ and a current position $v \in V_\circ$, the prescribed move is legal, *i.e* along an edge: $(v, \sigma(w \cdot v)) \in E$. Strategies for Adam are defined similarly, and usually denoted by τ. Once a game $\mathcal{G} = (\mathcal{A}, \Phi)$, a starting vertex v_0 and strategies σ for Eve and τ for Adam are fixed, there is a unique play denoted by $\pi(v_0, \sigma, \tau)$, which is said to be winning for Eve if it belongs to Φ. The sentence "Eve wins from v_0" means that she has a winning

strategy from v_0, that is a strategy σ such that for all strategy τ for Adam, the play $\pi(v_0, \sigma, \tau)$ is winning. The first natural problem we consider is to "solve the game", that is given a game $\mathcal{G}$ and a starting vertex v_0, to decide whether Eve wins from v_0. We denote by $\mathcal{W}_E(\mathcal{G})$ the winning positions of Eve, that is the set of vertices from where Eve wins (also referred as winning set), and analogously $\mathcal{W}_A(\mathcal{G})$ for Adam. We can prove that in generalized reachability games, we have $\mathcal{W}_E(\mathcal{A}, \Phi) \cup \mathcal{W}_A(\mathcal{A}, \Phi) = V$: from any vertex, either of the two players has a winning strategy. We say that the games are *determined*.

The strategies as defined in their full generality above are infinite objects. Indeed, in this general setting, to pick the next-move, Eve considers the whole history of the play, whose size grows arbitrarily. A nicer setting, giving rise to finitely-representable objects, is to define strategies relying on memory structures. Formally, a *memory structure* $\mathcal{M} = (M, m_0, \mu)$ for an arena $\mathcal{A}$ consists of a set M of memory states, an initial memory state $m_0 \in M$, and an update function $\mu : M \times E \to M$. A memory structure is similar in fashion to an automaton synchronized with the arena: it starts from m_0 and reads the sequence of edges produced by the arena. Whenever an edge is taken, the current state is updated using the update function μ. A strategy relying on a memory structure $\mathcal{M}$, whenever it picks the next move, considers only the current vertex and the current memory state: it is thus given by a next-move function $\nu : V_\circ \times M \to V$. Formally, given a memory structure $\mathcal{M}$ and a next-move function ν, we can define a strategy σ for Eve by $\sigma(w \cdot v) = \nu(v, \mu^*(w \cdot v))$. (The update function can be extended to a function $\mu^* : V^+ \to M$ by defining $\mu^*(v) = m_0$ and $\mu^*(w \cdot u \cdot v) = \mu(\mu^*(w \cdot u), (u, v))$.) A strategy with memory structure $\mathcal{M}$ has finite memory if M is a finite set. It is *memoryless*, or *positional* if M is a singleton: in this case, the choice for the next move only depends on the current vertex. Note that a memoryless strategy can be described as a function $\sigma : V_\circ \to V$.

We can make the synchronized product explicit: an arena $\mathcal{A}$ and a memory structure $\mathcal{M}$ for $\mathcal{A}$ induce the expanded arena $\mathcal{A} \times \mathcal{M} = (V \times M, (V_\circ \times M, V_\square \times M), E \times \mu)$ where $E \times \mu$ is defined by: $((v, m), (v', m')) \in E'$ if $(v, v') \in E$ and $\mu(m, (v, v')) = m'$. There is a natural one-to-one mapping between plays in $\mathcal{A}$ and in $\mathcal{A} \times \mathcal{M}$, and also from memoryless strategies in $\mathcal{A} \times \mathcal{M}$ to strategies in $\mathcal{A}$ using $\mathcal{M}$ as memory structure. It follows that if a player has a memoryless winning strategy for the arena $\mathcal{A} \times \mathcal{M}$, then he has a winning strategy using $\mathcal{M}$ as memory structure for the arena $\mathcal{A}$. This *key* property will be used later on.

A *reachability objective* requires that a vertex from a given subset of vertices F is reached: $\mathrm{Reach}(F) = \{v_0, v_1, v_2 \ldots \mid \exists p \in \mathbb{N}, v_p \in F\} \subseteq \Pi$. Games in the form $\mathcal{G} = (\mathcal{A}, \mathrm{Reach}(F))$ are called reachability games. To determine whether Eve wins a reachability game, we compute the reachability set attractor.

We define the sequence $(\mathrm{Attr}_i(F))_{i\geq 0}$:

$$\begin{array}{lll} \mathrm{Attr}_0(F) & = F & \\ \mathrm{Attr}_{i+1}(F) = \mathrm{Attr}_i(F) \cup & \{u \in V_\circ \mid \exists (u,v) \in E, v \in \mathrm{Attr}_i(F)\} \\ \cup & \{u \in V_\Box \mid \forall (u,v) \in E, v \in \mathrm{Attr}_i(F)\} \end{array}$$

Then $\mathrm{Attr}(F)$ is the limit of the non-decreasing sequence $(\mathrm{Attr}_i(F))_{i\geq 0}$. We can prove that $\mathcal{W}_E(\mathcal{A}, \mathrm{Reach}(F))$ is exactly $\mathrm{Attr}(F)$.

Generalized reachability objectives. A *generalized reachability objective* requires that each of the given k subsets of vertices $F_1, \ldots, F_k$ is reached:

$$\mathrm{GenReach}(F_1, \ldots, F_k) = \{\pi \mid \forall i, \exists p_i \in \mathbb{N}, v_{p_i} \in F_i\}.$$

Associating to each reachability objective a color, we can reformulate the generalized reachability objective: it requires to see each of the k colors at least once, in any order. Games in the form $\mathcal{G} = (\mathcal{A}, \mathrm{GenReach}(F_1, \ldots, F_k))$ are called generalized reachability games. The special cases where in $\mathcal{A}$, $V_\Box$ (respectively $V_\circ$) is empty are called one-player (respectively opponent-player) generalized reachability games.

Example 1. We consider the arena drawn in Figure 1. A generalized reachability game is defined by the objective $\mathrm{GenReach}(\{1,2\},\{3\})$. The central vertex is the initial one. Eve tries to visit one of the two thick vertices and the dashed vertex.

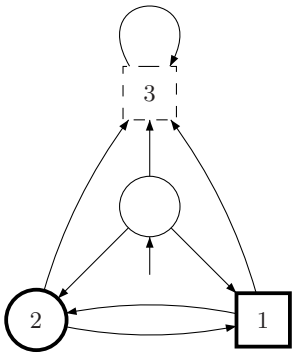

Fig. 1. An example of a generalized reachability game

Contributions. Our contributions are as follows:

– We first prove that deciding the winner in generalized reachability games is PSPACE-complete. Using the same ideas, we also show that the one-player restriction, where all vertices belong to Eve, is NP-complete, and that the opponent-player restriction, where all vertices belong to Adam, can be solved in polynomial time. On the positive side, it is fixed-parameter tractable (FPT) with the number k of colors as parameter.
– We study the size of the winning strategies for both players: we prove matching upper and lower bounds, *i.e* in any arena, if Eve has a winning strategy, then she has a winning strategy that uses $2^k - 1$ memory states, and there is an arena where Eve wins but there are no winning strategies with less than $2^k - 1$ memory states, and similarly for Adam with the bound $\binom{k}{\lfloor k/2 \rfloor}$.
– We then consider the subclasses where we restrict the number of vertices sharing the same color (in other words, the size of reachability sets). This reveals a trichotomy: if three vertices are allowed to share the same color, then deciding the winner is, as in the general case, PSPACE-complete. However, if each color appears only once, then the problem is polynomial. If each color appears only twice, then the problem is polynomial for one-player games, where Eve controls all vertices.

Outline. In section 3, we first study the complexity of solving generalized reachability games, for two-player and one-player games, and then give matching upper and lower bounds for the memory required. In section 4, we consider the subclasses of games where the size of reachability set is restricted, in order to find tractable subclasses.

3 The complexity of generalized reachability games

In this section we prove that the winner problem in generalized reachability games is PSPACE-complete. Our PSPACE-hardness result follows from a reduction from QBF (evaluation of a quantified boolean formula in conjunctive normal form). However, we show that solving generalized reachability games with few colors is easy, as it is fixed-parameter tractable using the number of colors as parameter.

We then study one-player restrictions. We prove that the one-player generalized reachability games are NP-complete. The other one-player restriction, opponent-player generalized reachability games, can be solved in polynomial time.

The last subsection investigates memory requirements for both players. We present matching upper and lower bounds: Eve needs $2^k - 1$ memory states and Adam $\binom{k}{\lfloor k/2 \rfloor}$, where k is the number of colors.

3.1 PSPACE-completeness of solving generalized reachability games

As a first step we define a reduction from QBF to the winner problem of generalized reachability games. Consider a quantified boolean formula

$$Q_1 x_1 \, Q_2 x_2 \, \ldots Q_n x_n \, \phi \, ,$$

where ϕ is a propositional formula in conjunctive normal form, *i.e*

$$\phi = \bigwedge_{i \leq k} \ell_{i,1} \, \vee \, \ell_{i,2} \, \vee \, \ldots \, \vee \, \ell_{i,j_i}$$

and $\ell_{i,j}$ is either x_i or $\neg x_i$ for some $i \leq n$. We construct a generalized reachability game where Eve wins if and only if the formula is true. Intuitively, the two players will sequentially choose to assign values to variables, following the quantification order and starting from the outermost variable. Eve chooses existential variables and Adam chooses universal variables. Formally, the game is as follows:

- for each variable x_i, there are two vertices, x_i and $\overline{x_i}$;
- for each variable x_i, there is a choice vertex v_i which leads to x_i and $\overline{x_i}$. The choice vertex belongs to Eve if x_i is existentially quantified, and to Adam if x_i is universally quantified;
- for each variable x_i with $i < n$, there are two edges from x_i and $\overline{x_i}$ to the next choice vertex v_{i+1};
- there is a sink s, and two edges from x_n and $\overline{x_n}$ to s;
- for each clause $\{\ell_{i,1}, \ldots, \ell_{i,j_i}\}$, there is a reachability objective F_i which contains the corresponding vertices;
- the generalized reachability objective is given by $\mathrm{GenReach}(F_1, \ldots, F_k)$.

The initial vertex is v_1. There is a natural bijection between assignments of the variables and plays in this game; and an assignment satisfies the formula ϕ if and only if the play satisfies the generalized reachability objective. The evaluation order of the variables being the same in the formula and in the game, we conclude that Eve has a winning strategy if and only if the formula is true.

Example 2. We consider the following quantified boolean formula

$$\forall x \, \exists y \, \forall z \, (x \vee \neg y) \wedge (\neg y \vee z) \, .$$

Figure 2 shows the game built by the reduction. The generalized reachability objective is $\mathrm{Reach}(\{x, \overline{y}\}) \wedge \mathrm{Reach}(\{\overline{y}, z\})$. Thick vertices represent the first reachability objective and dashed vertices the second one.

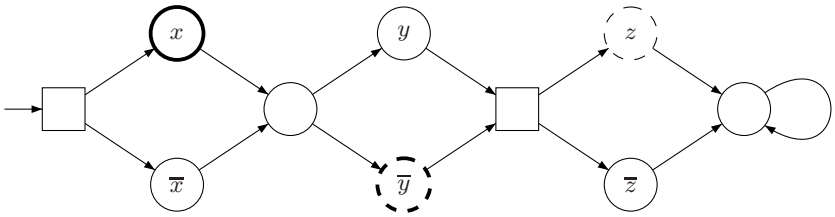

Fig. 2. An example of the reduction from QBF to generalized reachability games.

Theorem 1 (Complexity of generalized reachability games). *Solving generalized reachability games is* PSPACE*-complete.*

Proof. The previous reduction implies the PSPACE-hardness.

Let us first make a simple observation: if Eve has a winning strategy, then she has a winning strategy that visits each reachability set within $n \cdot k$ steps. Indeed, if she can enforce to visit a subset of vertices, then she can enforce it within n steps.

Relying on this remark, we can simulate the game for up to $n \cdot k$ steps using an alternating Turing machine: whenever a vertex belongs to Eve, the corresponding state is disjunctive, and it is conjunctive if the vertex belongs to Adam. A path of length $n \cdot k$ is accepted if it is winning, *i.e* if it contains one vertex from each reachability set F_i. This machine accepts if and only if Eve wins, and works in polynomial time. Since $\mathrm{APTIME} = \mathrm{PSPACE}$, the result follows. ∎

3.2 Parameterized complexity

Solving generalized reachability games with few colors is easy:

Theorem 2 (Generalized reachability games with k colors). *Solving generalized reachability games is fixed-parameter tractable (*FPT*) with the number of colors as parameter.*

Roughly speaking, the only information needed during a play is the subset of reachability sets already visited. We build a memory structure that keeps track of this information. By constructing the product with this memory structure, we turn a generalized reachability game into a (classical) reachability game.

Proof. We consider $\mathcal{G} = (G, \mathrm{GenReach}(F_1, \ldots, F_k))$ a generalized reachability game, and v_0 a starting vertex. The memory structure $\mathcal{M}$ is defined by $(2^{\{1,\ldots,k\}}, m_0, \mu)$, where m_0 is $\{i \mid v_0 \in F_i\}$, and $\mu(S, (v, v')) = S \cup \{i \mid v' \in F_i\}$. Let $F = \{(\_, S) \mid S = \{1, \ldots, k\}\}$: a play for the generalized reachability

game $\mathcal{G}$ from v_0 is winning if and only if it is winning for the reachability game $\mathcal{G} \times \mathcal{M} = (G \times \mathcal{M}, \mathrm{Reach}(F))$ from (v_0, m_0). Since deciding the winner in a reachability game can be done in linear time using an attractor computation, solving a generalized reachability game can be done in time $2^k \times O(n+m)$. ∎

3.3 Solving one-player restrictions

Theorem 3 (One-player restrictions). *Solving one-player generalized reachability games is* NP*-complete. Solving opponent-player generalized reachability games is polynomial.*

Proof. We first deal with one-player generalized reachability games, where Eve controls all vertices. In our previous reduction, consider the case where all variables in the original formula are quantified existentially. Then the problem corresponds to SAT (satisfiability of a boolean formula in conjunctive normal form), which is NP-complete. Resulting games are one-player games, *i.e* all vertices belong to Eve, hence solving one-player generalized reachability games is NP-hard.

We describe a non-deterministic algorithm to solve these games in polynomial time. As noted before, if Eve wins, then she has a winning strategy that wins within $n \cdot k$ steps. The algorithm guesses a path of length $n \cdot k$ and checks whether it is winning. It follows that solving one-player generalized reachability games is NP-complete.

We now consider opponent-player generalized reachability games, given by the objective $\mathrm{GenReach}(F_1, \ldots, F_k)$. The winning set for Adam is $V \setminus \bigcap_i \mathrm{Attr}(F_i)$, which can be computed in quadratic time. ∎

3.4 Memory requirements

We first present upper bounds:

Lemma 1 (Memory upper bounds). *For all generalized reachability games* $\mathcal{G} = (G, \mathrm{GenReach}(F_1, \ldots, F_k))$*,*

- *if Eve wins, then she wins using a strategy with memory* $2^k - 1$*;*
- *if Adam wins, then he wins using a strategy with memory* $\binom{k}{\lfloor k/2 \rfloor}$.

As in the proof for FPT membership, we make use of the memory structure $\mathcal{M} = (2^{\{1,\ldots,k\}}, m_0, \mu)$, where m_0 is $\{i \mid v_0 \in F_i\}$, and $\mu(S, (v, v')) = S \cup \{i \mid v' \in F_i\}$. Setting F as $\{(\_, S) \mid S = \{1, \ldots, k\}\}$, a play for the generalized reachability game $\mathcal{G}$ from v_0 is winning if and only if it is winning for the reachability game $\mathcal{G} \times \mathcal{M} = (G \times \mathcal{M}, \mathrm{Reach}(F))$ from (v_0, m_0).

Proof. We consider $\mathcal{G} = (G, \mathrm{GenReach}(F_1, \ldots, F_k))$ a generalized reachability game, and v_0 a starting vertex. Since in the reachability game $\mathcal{G} \times \mathcal{M}$, each player has memoryless winning strategies, each player has in $\mathcal{G}$ a winning strategy using $\mathcal{M}$ as memory structure.

The memory set of $\mathcal{M}$ has size 2^k. In order to get the correct bounds for each player, we rely on two observations.

- Eve does not need a specific memory state to remember that all colors have been reached, as in this case, she has already won. Thus, she can always win with $2^k - 1$ memory states.
- If Adam wins in $\mathcal{G} \times \mathcal{M}$ from (v, S) and $S' \subseteq S$, then he wins from (v, S') *using the same strategy*. Consider $v \in V$ a vertex in $\mathcal{G}$, and the set of subsets S such that (v, S) belongs to Adam's winning set. Its maximal (with respect to inclusion) elements are incomparable, so there are at most $\binom{k}{\lfloor k/2 \rfloor}$, we denote them by $S_1(v), \ldots, S_p(v)$. The idea is that from v, there are only p different options Adam has to consider, namely $S_1(v), \ldots, S_p(v)$. Indeed, for any S such that (v, S) is winning for Adam, there exists an i such that $S \subseteq S_i(v)$, so Adam can forget S and assume the current position is $(v, S_i(v))$.
  We define a memory structure on the memory set $\{1, \ldots, \binom{k}{\lfloor k/2 \rfloor}\}$. We aim at constructing a strategy that will ensure that after a finite play $\pi \cdot v$, the memory state is an i such that $S_i(v)$ contains the set of visited colors. If the initial vertex is v_0, the initial memory state is an i_0 such that $S_{i_0}(v_0)$ contains m_0. We define the update function: $\mu(i, (v, v'))$ is a j such that $S_j(v')$ contains $\mu(S_i(v), v') = S_i(v) \cup \{i \mid v' \in F_i\}$.
  Let us turn to the next-move function. Consider (v, S) in Adam's winning set, then there exists a transition to some (v', S') also in Adam's winning set. Applying this to $(v, S_i(v))$ such that $S \subseteq S_i(v)$, we get a vertex v', and define $\nu(v, i)$ to v'. Playing this strategy, the above invariant is satisfied, and thus ensures to stay forever in Adam's winning set, so it is winning. The memory set contains $\binom{k}{\lfloor k/2 \rfloor}$ memory states.

∎

Lemma 2 (Memory lower bounds for both players). *For all k,*

- *there exists $\mathcal{G} = (G, \mathrm{GenReach}(F_1, \ldots, F_k))$ a generalized reachability game, where Eve needs $2^k - 1$ memory states to win;*
- *there exists $\mathcal{G} = (G, \mathrm{GenReach}(F_1, \ldots, F_k))$ a generalized reachability game, where Adam needs $\binom{k}{\lfloor k/2 \rfloor}$ memory states to win.*

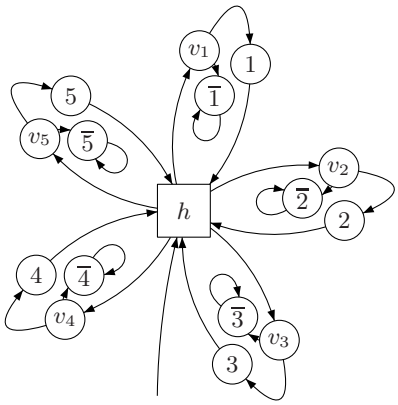

Fig. 3. A generalized reachability game where Eve needs $2^k - 1$ memory states to win

Proof. We first describe a generalized reachability game where Eve needs $2^k - 1$ memory states to win. This example was proposed in [CHH11] in a similar framework. The arena is shown in Figure 3, for $k = 5$. A vertex labelled by i belongs to F_i, and a vertex labelled by $\overline{i}$ has all colors but i. A play starts from the heart h; first Adam chooses a petal i, then Eve chooses either to reach color i before going back to the heart (the play goes on), or to reach every colors but i and to stop the play. Eve wins with the following strategy: the first time Adam chooses the petal i, she goes back to the heart; the second time, she stops the play. This strategy uses 2^k memory states. She can save one memory state by dropping the memory state corresponding to the case where she saw each petal, as it is winning for her. However, we show that there is no winning strategy for Eve with less than $2^k - 1$ memory states. Let σ a strategy using the memory structure $\mathcal{M}$ with less than $2^k - 1$ memory states, and ν its next-move function. For each memory state m, we consider $S_m = \{i \mid \nu(v_i, \mu(m, (h, v_i))) = \overline{i}\}$, the set of petals where Eve would stop the play if Adam chose them. As there are less than $2^k - 1$ memory states, there is a strict subset X of $\{1, \ldots, k\}$ which is not the stopping set of any memory state. Adam can win against σ by choosing, at each step, a petal in the symmetric difference of X and S_m, where m is Eve's current memory under σ. (Indeed, if Adam plays forever in X, then Eve will never stop the play and only colors from X will be reached, otherwise, whenever Eve stops the play, the last memory state from the heart was an m such that $X \subset S_m$, and the petal chosen is an i that belongs to $S_m \setminus X$, hence that has never been reached.)

We now describe a generalized reachability game won by Adam, where he needs $\binom{k}{\lfloor k/2 \rfloor}$ memory states to win. Let $k = 2p + 1$. A play consists in three steps: first Eve chooses p colors, then Adam chooses p colors, and third Eve

chooses p colors. In order to win, Adam must visit exactly the same colors Eve visited (which requires $\binom{k}{\lfloor k/2 \rfloor}$ memory states), otherwise at least $p+1$ colors have been visited when Eve plays for the second time, and she can choose and visit the remaining colors that have not yet been visited. ∎

4 Restrictions on the size of reachability sets

The above section shows two different directions which make generalized reachability games hard: the first is the complexity of solving generalized reachability games (PSPACE-complete), and the second is the memory required to construct winning strategies for both players (exponential in the number of colors).

In this section, we restrict the size of the reachability sets in order to find tractable subclasses of generalized reachability games.

Notice that our reduction from QBF only implies PSPACE-hardness when reachability sets have size at least three. Indeed, note that in the reduction, the size of a reachability set in the generalized reachability game corresponds to the size of the corresponding clause of the formula. Since the problem of evaluating a quantified boolean formula is polynomial if the formula has two variables per clause, our reduction does not imply the PSPACE-hardness of solving generalized reachability games with reachability sets of size one or two. This remark motivates our study of the subclasses of generalized reachability games where each color appears once, and then where each color appears twice.

4.1 Reachability sets of size one

The case where reachability sets are singletons is polynomial:

Theorem 4 (Generalized reachability games where reachability sets have size 1). *Solving generalized reachability games where reachability sets are singletons is in* PTIME.

Proof. We denote by v_i the only vertex in F_i, for all i. In this case, the generalized reachability objective can be expressed by $\bigwedge_{i \leq k} \mathrm{Reach}(v_i)$. We will see that Eve wins if and only if the preorder defined by $v \preceq v'$ if $v \in \mathrm{Attr}(v')$ is total. Intuitively, it means that a winning strategy prescribes: "reach $v_{f(1)}$, then $v_{f(2)}$, and so on", where f is a permutation over $\{1, \ldots, k\}$.

We consider two cases:

– If the preorder $\preceq$ is total over $\{v_i \mid 1 \leq i \leq k\}$, then we show that $\mathcal{W}_E$, set of winning positions for Eve, is $\cap_i \mathrm{Attr}(v_i)$. Let $v \in \cap_i \mathrm{Attr}(v_i)$ and f a permutation over $\{1, \ldots, k\}$ such that for all $1 \leq i \leq k-1$, we have

$v_{f(i)} \in \mathrm{Attr}(v_{f(i+1)})$, we construct a winning strategy from v that reaches $v_{f(1)}$, then $v_{f(2)}$, and so on. Note that this strategy only needs k memory states. Conversely, if $v \notin \cap_i \mathrm{Attr}(v_i)$, then Eve cannot win, as Adam can prevent her from reaching some reachability set.

– If the preorder $\preceq$ is not total, then there exist v_i and v_j such that $v_i \notin \mathrm{Attr}(v_j)$ and $v_j \notin \mathrm{Attr}(v_i)$. In this case Adam wins from everywhere, following the strategy "if v_i or v_j has been reached, then avoid the other". Note that this strategy only needs 2 memory states.

Checking that the preorder $\preceq$ is total can be done in polynomial time. ∎

Note that as a corollary, we get memory upper bounds in this case: Eve needs at most k memory states and Adam at most 2. It is not difficult to see that these bounds are tight.

4.2 Reachability sets of size two

Let us now turn to the case where reachability sets have size two. We first extend the technique used for the previous case: it was stated that "Eve wins if and only if there is a total order on colored vertices". A similar approach works for one-player arenas, through a reduction to the satisfiability problem of boolean formulas where clauses have size two. (This latter problem is known to be decidable in polynomial time.)

Theorem 5 (Generalized reachability one-player games where color appears twice). *Solving generalized reachability one-player games where reachability sets have size two is in* PTIME.

Proof. As in the previous subsection, we consider the preorder defined by $v \preceq v'$ if $v \in \mathrm{Attr}(v')$. Note that in the case of one-player arenas, $v \in \mathrm{Attr}(v')$ reduces to "there is a path from v to v'".

Let $F_i = \{x_i, y_i\}$ be the reachability sets, and v_0 be a starting vertex. We assume without loss of generality that there is a path from v_0 to every F_i (that is, either to x_i or y_i), otherwise Eve cannot win. (This property is easily checked in deterministic polynomial time.) A first statement is as follows: Eve wins from v_0 if and only if there exist $v_1, \ldots, v_k$ colored vertices such that

1. for all $0 \leq i \leq k-1$, $v_i \preceq v_{i+1}$ and
2. each color appears in $\{v_1, \ldots, v_k\}$.

We turn this condition into a boolean formula where clauses have size 2. We consider the $2 \cdot k$ variables X_i and Y_i, that correspond to vertices x_i and y_i. We

define the formula ϕ:

$$\underbrace{\bigwedge\{(\neg X \vee \neg Y) \mid \text{ if } x \not\preceq y \text{ and } y \not\preceq x\}}_{(a)} \wedge \underbrace{\bigwedge_{i}(X_i \vee Y_i)}_{(b)},$$

where x, y ranges over colored vertices (that is, vertices from F_i for some i).

We argue that Eve wins from v_0 if and only if ϕ is satisfiable. Assume Eve wins from v_0: let $v_1, \ldots, v_k$ as in the previous statement, and set the corresponding variables to true and the others to false, we claim that the formula ϕ is satisfied. Indeed, condition 2. ensures that the clauses under-braced (b) are satisfied, and for the clauses under-braced (a), let x, y such that $x \not\preceq y$ and $y \not\preceq x$, if x is one of the v_i's, then y cannot be, so $\neg X \vee \neg Y$ holds. Conversely, assume that ϕ is satisfiable. The clauses under-braced (a) ensures that the order $\preceq$ is total over vertices set to true. The clauses under-braced (b) ensures that at least one vertex from each reachability set is set to true. Combining those two statements, we reach the condition stated above.

The latter allows to decide in polynomial time whether Eve wins from v_0 by checking the formula ϕ for satisfiability. ∎

We do not know the exact complexity of generalized reachability games where reachability sets have size 2. In the remaining of this subsection, we discuss this question, focusing on memory requirements for both players.

The memory required for Eve is still exponential, as shown in Figure 4 for $k = 4$. Specifically, it shows a generalized reachability game where reachability sets have size 2 won by Eve, where she needs $2^{\lfloor k/2 \rfloor + 1} - 1$ bits of memory to win. The arena is divided into two parts: the left hand side is a flower with $\lfloor k/2 \rfloor$ petals, and the right hand side a one-player arena. The game starts at the heart of the flower. First, Eve asks for each petal a color. Once this task is completed, she can move to the right hand side to reach the remaining colors. Eve needs to remember the $\lfloor k/2 \rfloor$ choices made by Adam (one for each petal), in order to reverse them: if Adam chose the color 1, then the color 2 has not been reached, so Eve has to choose color 2. Remembering those choices and asking for each petal requires $2^{\lfloor k/2 \rfloor + 1} - 1$ memory states. (This is the size of the complete binary tree of depth $\lfloor k/2 \rfloor$.)

On the other hand, the exact memory required for Adam remains open. The figure 5, following an idea of Christof Loeding, shows a generalized reachability game where reachability sets have size 2 won by Adam, where he needs 4 memory states to win. The game starts from the left hand side vertex. First Eve chooses and visits three of the four colors (two colors in the first column, 1 and 2 or 3 and 4, and then one in the second column), and sends the pebble

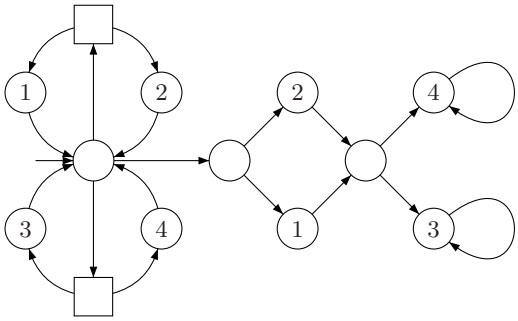

Fig. 4. A generalized reachability game where Eve needs $2^{\lfloor k/2 \rfloor + 1} - 1$ memory states to win

to the right hand side vertex, controlled by Adam. There, he has four options, each allowing all colors but one. Remembering the four possibilities requires four memory states, and leads to a win. However, with less memory states, one of the four option will never be played, and Eve wins.

Quite surprisingly, we could not generalize this example to obtain a better lower bound than 4. We do not know whether this bound is tight (in any arena, if Adam wins, then he has a winning strategy with 4 memory states), which is plausible. Note that this would imply a $\mathrm{coNP}^{\mathrm{NP}}$ algorithm: guess a winning strategy for Adam with 4 memory states, and compose this strategy with the game, then solve the resulting one-player game.

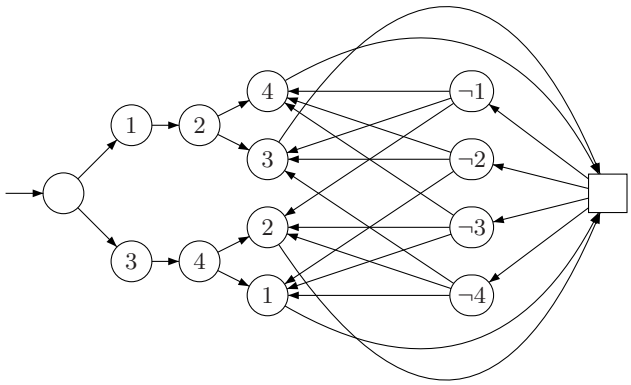

Fig. 5. A generalized reachability game where Adam needs 4 memory states to win

Open problems. We were not able to give the exact complexity of generalized reachability games where reachability sets have size 2. The memory approach,

showing that Adam has winning strategy of constant size, seems promising towards this question.

References

[CHH11] Krishnendu Chatterjee, Thomas A. Henzinger, and Florian Horn. The Complexity of Request-response Games. In *LATA*, pages 227–237, 2011.

[GTW02] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *LNCS*. Springer-Verlag, 2002.

[HTW08] Florian Horn, Wolfgang Thomas, and Nico Wallmeier. Optimal Strategy Synthesis in Request-Response Games. In *Proceedings of the 6th International Symposium on Automated Technology for Verification and Analysis, ATVA'08*, volume 5311 of *LNCS*, pages 361–373. Springer-Verlag, 2008.

[KPV07] Orna Kupferman, Nir Piterman, and Moshe Y. Vardi. From Liveness to Promptness. In *Proceedings of the 19th International Conference on Computer Aided Verification, CAV'07*, volume 4590 of *LNCS*, pages 406–419. Springer-Verlag, 2007.

[KVW00] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, 2000.

[Mos91] Andrzej Wlodzimierz Mostowski. Hierarchies of weak automata and weak monadic formulas. *Theoretical Computer Science*, 83(2):323–335, 1991.

[NSW02] Jakub Neumann, Andrzej Szepietowski, and Igor Walukiewicz. Complexity of weak acceptance conditions in tree automata. *Information Processing Letters*, 84(4):181–187, 2002.

[SW74] Ludwig Staiger and Klaus W. Wagner. Automatentheoretische und automatenfreie charakterisierungen topologischer klassen regulärer folgenmengen. *Elektronische Informationsverarbeitung und Kybernetik*, 10(7):379–392, 1974.

[TBG09] Mathieu Tracol, Christel Baier, and Marcus Größer. Recurrence and transience for probabilistic automata. In *International Conference on the Foundations of Software Technology and Theoretical Computer Science, FSTTCS'09*, pages 395–406, 2009.

[Zim11] Martin Zimmermann. Optimal bounds in parametric LTL games. In *International Symposium on Games, Automata, Logics and Formal Verification, GandALF'11*, pages 146–161, 2011.